Paper No. 20

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte GERRIT H. SOEPENBERG
and RONALD M. TOL

Appeal No. 2004-0849
Application No. 09/329,391

ON BRIEF

Before HAIRSTON, GROSS, and LEVY, Administrative Patent Judges.
LEVY, Administrative Patent Judge.

DECISION ON APPEAL

This is a decision on appeal under 35 U.S.C. § 134 from the examiner's final rejection of claims 1-8, which are all of the claims pending in this application.

BACKGROUND

Appellants' invention relates to a transmission system for transmitting multiplex signals from a transmitter to a receiver. The multiplex signal comprises a periodically repeated plurality of modules, with each module comprising at least one object

(specification, page 1). Extracting means extract the objects in dependence on module related information present in the multiplex signal (specification, page 2). An understanding of the invention can be derived from a reading of exemplary claim 1, which is reproduced as follows:

> 1. A transmission system for transmitting a multiplex signal from a transmitter to a receiver, said multiplex signal comprising a periodically repeated plurality of modules each comprising at least one object, the receiver comprising extracting means for extracting objects from the multiplex signal, wherein the extracting means are embodied so as to extract the objects in dependence on module related information present in the multiplex signal.

The prior art reference of record relied upon by the examiner in rejecting the appealed claims is:

Wasilewski	5,420,866	May 30, 1995

Claims 1-8 stand rejected under 35 U.S.C. § 102(b) as being anticipated by Wasilewski.

Rather than reiterate the conflicting viewpoints advanced by the examiner and appellants regarding the above-noted rejection, we make reference to the examiner's answer (Paper No. 16, mailed April 22, 2003) for the examiner's complete reasoning in support of the rejection, and to appellants' brief (Paper No. 15, filed April 3, 2003) and reply brief (Paper No. 17, filed June 26, 2003) for appellants' arguments thereagainst. Only those

arguments actually made by appellants have been considered in this decision. Arguments which appellants could have made but chose not to make in the brief have not been considered.

OPINION

In reaching our decision in this appeal, we have carefully considered the subject matter on appeal, the rejection advanced by the examiner, and the evidence of anticipation relied upon by the examiner as support for the rejection. We have, likewise, reviewed and taken into consideration, in reaching our decision, appellants' arguments set forth in the briefs along with the examiner's rationale in support of the rejection and arguments in rebuttal set forth in the examiner's answer.

Upon consideration of the record before us, we affirm. We observe at the outset appellants' statement (brief, page 4) that the claims may be grouped together for purposes of this appeal. Consistent with this assertion, appellants' arguments are directed to claim 1. Accordingly, we select claim 1 as representative of the group.

To anticipate a claim, a prior art reference must disclose every limitation of the claimed invention, either explicitly or inherently. In re Schreiber, 128 F.3d 1473, 1477, 44 USPQ2d

1429, 1431 (Fed. Cir. 1997). As stated in In re Oelrich, 666 F.2d 578, 581, 212 USPQ 323, 326 (CCPA 1981) (quoting Hansgirg v. Kemmer, 102 F.2d 212, 214, 40 USPQ 665, 667 (CCPA 1939)) (internal citations omitted):

> Inherency, however, may not be established by probabilities or possibilities. The mere fact that a certain thing may result from a given set of circumstances is not sufficient. If, however, the disclosure is sufficient to show that the natural result flowing from the operation as taught would result in the performance of the questioned function, it seems to be well settled that the disclosure should be regarded as sufficient.

Thus, a prior art reference may anticipate when the claim limitation or limitations not expressly found in that reference are nonetheless inherent in it. See In re Oelrich, 666 F.2d at 581, 212 USPQ at 326; Verdegaal Bros., Inc. v. Union Oil Co., 814 F.2d 628, 630, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). Under the principles of inherency, if the prior art necessarily functions in accordance with, or includes, the claimed limitations, it anticipates. See In re King, 801 F.2d 1324, 1326, 231 USPQ 136, 138 (Fed. Cir. 1986).

Turning to claim 1, appellants assert (brief, page 7) that the term "objects" as used in the claims, refers to self contained executable code segments. Appellants argue (id.) that they are entitled to define the meaning of claim terms as long as the meaning assigned to the term is not repugnant to the term's

well known usage. It is further argued (brief, pages 7 and 8) that because appellants' definition of the term object is not repugnant to the term's well known usage, the examiner cannot ignore the appellants' definition.

From our review of appellants' specification, we find that the specification states (page 4)that "[t]he response time of this kind of applications can be improved considerably by . . . pre-fetching and storing data sections or objects in the set-top box 14 by pre-fetching and storing data sections or objects." The use of "data sections" in contrast to "objects" supports appellants' argument that objects refer to the use of self-contained executable code segments. However, in the next paragraph, the specification states that "[t]hese objects are to be presented to the user in some manner, e.g., a *viewer* application presents a front page representing the table of contents with links to other elements of the newspaper." From this disclosure in appellants' specification, we find that because an application presents the front page, data in the front page can be objects. In addition, the specification (page 5), states that the "electronic newspaper application will construct a *filter* . . . a filter can specify that all objects related to sports, financials or headlines should be pre-fetched (by the

extracting means 16) and cached on the storaged device 19." This passage of the specification discloses that "objects" can be blocks of data that represent the sports, financial or headline sections of a newspaper. We find from these disclosures, that the term "objects" does not necessarily refer to "self-contained executable code segments" as advanced by appellants, but could also refer to a passive entity that contains data, such as the words that make up the headline portion of a newspaper.

We find this to be consistent with the definition of an object as set forth in the IBM Dictionary of Computing[1] which defines "object" as:

> A passive entity that contains or receives data; for example, bytes, blocks, clocks, fields, files, directories, displays, keyboards, network nodes, pages, printers, processors, programs, records, segments, words. Access to an object implies access to the information it contains.

From this, we find that the term "objects," as defined in appellants' specification, is not limited to objects being executable code segments. Because the term objects, as defined in appellants' specification, is not limited to being self-contained executable code segments, we find that the term objects, as set forth in claim 1 should not be given the narrow

---

[1] © 1994. A copy of the pertinent page is enclosed with the decision.

construction of "self-contained executable code segments" as advanced by appellants.

We agree with appellants (brief, page 7) that the examiner has rejected appellants' definition of an object. The IEEE article relied upon by the examiner for a definition of an object does not support the examiner's position that the definition refers to an object as simply data. Rather, the Dictionary definition refers to an object as "a piece of data defined by the operations performed on it," as noted by appellants (reply brief, page 2). However, in view of the broader definition of objects found in the IBM Dictionary of Computing, which we have found to be consistent with the definition of the term "objects" as set forth in appellants' specification, we find that the ECMs of Wasilewski comprise objects.

Wasilewski teaches that the multiplexed data stream contains a block of data known as Entitlement Control Messages (ECM) which contain encryption or scrambling related information for one of the elementary streams in a transport stream (col. 4, lines 7-15). Wasilewski further discloses that an ECM may be used to transmit the encryption control words necessary for decrypting a particular elementary stream, and that the ECMs may be

transmitted in dedicated transport packets for transmission to decoders (col. 4, lines 15-20 and col. 11, lines 43-48). Wasilewski further discloses (col. 9, lines 39-41) that encryption related information must be provided to decoders in order to enable the decoders to decrypt or descramble the encrypted data. Wasilewski additionally discloses (col. 14, lines 13-25) that:

> The ECMs for a given elementary stream contain encryption related information necessary for decrypting that elementary stream. Once the demultiplexer 116 has identified the PIDs of the Transport Packets that contain the ECMs for each respectively elementary stream of the selected program, the demultiplexer extracts those Transport Packets as they are received in the incoming Transport Stream. As the Transport Packets that contain the data for each elementary stream of the selected program are retrieved from the incoming Transport Stream, they are provided, in sequence, to a decryption/descrambling and program authorization unit 120.

From the disclosure of Wasilewski, we find that the ECMs are objects that are transported in Transport packets, which we find to be modules. In addition, we find that the transport packets 6 are multiplexed into transport streams (col. 2, lines 31-33). In sum, we find that the multiplex signal includes Transport Packets, and that Transport Packets containing ECMs meet the claimed modules comprising at least one object. Furthermore, we

find that the decoding means meets the claimed extracting means as the decoding means decodes the incoming transport stream, that the Transport Stream meets the claimed multiplex signal, and that the ECMs are extracted in dependence on module related information (at least the PIDs) for the ECMS that are present in the multiplex signal. In addition, we agree with the examiner that program definition 2, includes module 72. We find that as shown in figure 4, program number 74, other information 76, elementary PID count 78, and elementary stream definitions 80 comprise a module. We further agree with the examiner that because an object, as disclosed by appellants, is not limited to executable code segments, the elementary stream definitions 80 are objects in a program definition or module 72. Because each elementary stream definition includes an elementary stream PID, the PID for the ECM, and other information, the elementary stream definition is an object. In addition, we find that because a Packet ID (PID) is assigned to each elementary stream, and the PID is inserted into the headers for each Transport Packet (col. 1, lines 54-56 and col. 2, lines 26-30), and because a PID is provided for each Transport Packet including an ECM (figure 3B), the PIDs are objects.

Appellants assert (brief, page 8) that there is no disclosure, teaching or suggestion in Wasilewski that the program definitions in the program map table are periodically repeated. It is argued (id.) that the examiner has not shown that the same program map table or the same program definitions are communicated periodically to decoders in the system and that Wasilewski may only send the program map table to a decoder only once, and there would be no need to send additional transmissions of the program map table to the decoder.

From our review of Wasilewski, we agree with the examiner (answer, page 8) that the program map table (list of TV programs) is inherently periodically transmitted because if a program map table includes objects representing TV programs available for viewing, the program map table will at least in some instances be repeated (see col. 10, lines 12-25). The fact that a program map table may be sent to a decoder only once in a particular situation does not mean that the program map table is not repeated in other situations, as discussed, supra. We are cognizant that Wasilewski does not disclose the use of a carousel, which provides cyclical playout of data files over a

specific time frame[2]. However, we observe that the carousel is not claimed. Appellants could have chosen to claim the carousel during prosecution before the examiner, but have not done so. Similarly, appellants could have specifically defined the term object in their claims as executable code, but have not done so. As broadly claimed, the recitation "said multiplex signal comprising a periodically repeated plurality of modules each comprising at least one object" is broad enough to read on a repeated transmission of sports scores, headline news, or a listing of the TV shows available for viewing at a particular time. Accordingly, we find that this limitation is met by Wasilewski.

Appellants further assert (brief, page 9) that claim 1 recites that extracting means extracts objects in dependence on module related information present in the multiplex signal, and that:

> *Wasilewski* contains no mention that the program number is used to extract the elementary stream definitions from the program map table. Instead, the program number identifies the program (such as a television program) in the MPEG-2 stream. (Col. 10, Lines 41-43). There is no mention in *Wasilewski* that the elementary stream definitions are extracted using the program number.

---

[2] See page 837 of IEEE article "The Set-Top Box as 'Multi-Media Terminal,'" which is of record in the application.

As we found, supra, the quoted limitation is met by the disclosure of Wasilewski that the ECMs are extracted based on information, such as the ECM PIDs present in the multiplex signal. In addition, from our review of Wasilewski, we are in agreement with the examiner, for the reasons set forth in the answer, that:

> Wasilewski teaches transmitting a Program Map Table including user selectable program numbers (col. 2 lines 63-67 and col. 3 lines 1-22). Those numbers (for example Channel 4) identify elementary streams Packet IDs (Channel 4 video and Channel 4 audio) associated with the selected program number. The user does not know the elementary streams Packet IDs 'making-up' the elementary streams of Channel 4, so the program number (4) is used to identify and extract the stream Packet IDs from the Program Map, so that the latter can be used to isolate the data packets forming the program stream.

In addition, we find further support for the examiner's position on page 835 of the IEEE article "The Set-Top Box as 'Multi-Media Terminal,'" which states "Program Map Table (PMT) provides the mapping between program numbers and the program elements that comprise them, which is the complete collection of all program definitions for a transport stream" (underlining added).

We are not persuaded by appellants' assertion (reply brief, page 3) that "[w]hen the stream Packet IDs are retrieved using

the program number, the Program Map Table has already been received and stored by the receiver. In other words, the stream Packet IDs are being retrieved from a table stored in memory at the receiver, rather than being extracted from a 'multiplex signal'." In our view, even if the program map table has already been extracted from the transport stream and stored by the receiver when the stream Packet IDs are retrieved using the program numbers, we find no language in the claim, and none has been brought to our attention by appellants, that would preclude information from the received transport stream from being extracted from the incoming signal and stored in a memory before retrieving the Packet IDs using the program numbers. Claim 1 requires that the extracting means extract objects from the multiplex signal; i.e., as broadly drafted, nothing in the claim precludes the extracting and storing of the program map table as part of the process of extracting objects in dependence on module related information present in the multiplex signal.

From all of the above, we find that Wasilewski anticipates claim 1. Accordingly, the rejection of claim 1 under 35 U.S.C. § 102(b) is affirmed. As claims 2-8 fall with claim 1 (brief, page 4), the rejection of claims 2-8 under 35 U.S.C. § 102 (b) is affirmed.

CONCLUSION

To summarize, the decision of the examiner to reject claims 1-8 under 35 U.S.C. § 102(b) is affirmed.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 CFR § 1.136 (a).

AFFIRMED

KENNETH W. HAIRSTON
Administrative Patent Judge

ANITA PELLMAN GROSS
Administrative Patent Judge

STUART S. LEVY
Administrative Patent Judge

BOARD OF PATENT APPEALS AND INTERFERENCES

SSL/kis

PHILIPS INTELLECTUAL PROPERTY & STANDARDS
P. O. BOX 3001
BRIARCLIFF MANOR, NY 10510

IBM Dictionary of Computing

▼ The most comprehensive computing dictionary ever published

▼ More than 18,000 entries

Limitation of Liability

While the Editor and Publisher of this book have made reasonable efforts to ensure the accuracy and timeliness of the information contained herein, neither the Editor nor the Publisher shall have any liability with respect to loss or damage caused or alleged to be caused by reliance on any information contained herein.

7 8 9 0 DOC/DOC 9 9 8

ISBN 0-07-031488-8 (HC)
ISBN 0-07-031489-6 (PBK)

The sponsoring editor for this book was Daniel A. Gonneau and the production supervisor was Thomas G. Kowalczyk.

Printed and bound by R. R. Donnelley & Sons Company.

Tenth Edition (August 1993)

This is a major revision of the *IBM Dictionary of Computing,* SC20-1699-8, which is made obsolete by this edition. Changes are made periodically to the information provided herein.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country. Comments may be addressed to IBM Corporation, Department E37/656, P. O. Box 12195, Research Triangle Park, NC 27709.

International Edition

Copyright © 1994 by International Business Machines Corporation. Exclusive rights by McGraw-Hill, Inc. for manufacture and export. This book cannot be re-exported from the country to which it is consigned by McGraw-Hill. The International Edition is not available in North America.

When ordering this title, use ISBN 0-07-113383-6.

This book is printed on acid-free paper.

O

OA Office automation. See automated office.

OACBRU Open ACB request/response unit.

OAF Origin address field.

OAF' Origin address field prime.

OAM Storage Management Component In the Object Access Method, the component that determines where objects should be stored, manages object movement within the object storage hierarchy, and manages expiration attributes based on the installation storage management policy.

OAR Operator authorization record.

object (1) In computer security, anything to which access is controlled; for example, a file, a program, an area of main storage. (2) A passive entity that contains or receives data; for example, bytes, blocks, clocks, fields, files, directories, displays, keyboards, network nodes, pages, printers, processors, programs, records, segments, words. Access to an object implies access to the information it contains. (3) In SAA Common User Access architecture, something that a user works with to perform a task. Text and graphics are examples of objects. (4) In AIX Enhanced X-Windows, a software abstraction consisting of private data and private and public routines that operate on the private data. Users can interact with an object only through calls to the public routines of the object. (5) In the AIX object data manager, an instance or member of an object class, conceptually similar to a structure that is a member or array of structures. See also object class. (6) In programming languages, a data object. (7) In AIX graphics, synonym for display list. (8) In the Network Computing System, an entity that is manipulated by well-defined operation; for example, a disk, a file, a printer. Every object has a type and is accessed through an interface. (9) In the IBM ImagePlus system, a collection of structured fields. The first structured field provides a begin-object function and the last structured field provides an end-object function. The object may contain one or more other structured fields whose content consists of one or more data elements of a particular data type. An object may be assigned a name, which may be used to reference the object. Examples of objects are text, font, graphics, image, and formatted data objects. (10) In object-oriented design or programming, an abstraction consisting of data and the operations associated with that data. See also class. (11) In the AS/400 system, a named storage space that consists of a set of characteristics that describe itself and, in some cases, data. An object is anything that exists in and occupies space in storage and on which operations can be performed; for example, programs, files, libraries, and folders. (12) In SQL, anything that can be created or manipulated with SQL statements, such as databases, tables, views, or indexes. (13) See arithmetic object, compound object, data object, integral object.

Object Access Method (OAM) In the IBM ImagePlus system, a program that provides object storage, object retrieval, and object storage hierarchy management. The Object Access Method isolates applications from storage devices, storage management, and storage device hierarchy management.

object-action In SAA Common User Access architecture, a process sequence in which a user selects an object and then selects an action to apply to that object. Contrast with action-object.

object authority (1) In the AS/400 system, a specific authority that controls what a system user can do with an entire object; for example, object authority includes deleting, moving, or renaming an object. There are three types of object authorities: object existence, object management, and object operational. (2) The right to use or control an object. See also data rights, object rights.

object class A categorization or grouping of objects that share similar behaviors and circumstances.

object code Output from a compiler or assembler which is itself executable machine code or is suitable for processing to produce executable machine code. (A)

object code compatibility (1) Pertaining to object programs that can be run on two or more systems without recompilation or reassembly. (2) Contrast with source code compatibility.

OBJECT-COMPUTER In COBOL, the name of an Environment Division paragraph in which the computer environment, within which the object program is executed, is described.

object-computer entry In COBOL, an entry in the OBJECT-COMPUTER paragraph of the Environment Division that contains clauses that describe the computer environment in which the object program is to be executed.